# Technical Answers for Real World Problems (TARP) (CSE3999)

## FALL 2020-2021

**SARTHAK SURANA 17BCE0996**

**DHRUV MITTAL 17BCE2110**

# TITLE- DOCTOR'S ASSISSTANT SYSTEM

Slot- TF1

Under the guidance of

## Prof. ARUN KUMAR G

SCOPE, VIT, Vellore


VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

# Table of Contents



# Table of Figures

# ABSTRACT

Presently healthcare is the most important domain across the globe. Amidst this pandemic, all the doctors are very occupied and extremely busy. Getting proper treatment from doctor is getting difficult due to unavailability of doctors. Moreover, if a person feels uneasy and doesn't know what exactly the problem is, they really don't know which doctor they should take an appointment with. So, in this paper we intend to provide a web application as an all in one solution with an integrated voice assistant which can help you with the process of disease diagnosis. For those doctors and patients who may need a second brain to make sure the diagnosis is correct, this is a platform with an AI-based medical voice assistant. It can work verbally with the doctor / patient and can assist him/her with the diagnostic process. This voice assistant should be able to select a patient's illness on a confidence score to support diagnosis operations. Such a software can help both physicians and patients. Depending on the predicted disease/illness, the doctor may give the patient an e-prescription using the help of a voice assistant. The patient can also order medication using a voice assistant. The archives are stored in an orderly fashion, so users do not have to worry about losing them. This application is exclusively made to assist doctors and patients with intention of giving medical care in an interactive, life-saving and resource saving manner. This application is a minor step in achieving a bigger target of completely digitalizing our medical care.

# INTRODUCTION

Self-diagnosis is highly susceptible to error and is often extremely dangerous if wrong decisions are made on the basis of a wrong diagnosis. After looking at these misdiagnosis and adverse treatment practices by staff, we considered building a web system for the well-being of our community. This web app can help doctors and patients alike to store Electronic medical records, help in the process of disease diagnosis, fasten the process of prescription generation, aid in buying prescription medicines from 3rd party websites and much more. The interface of these healthcare-based applications is required to be easily usable by everyone. So, keeping this in mind we have integrated a voice assistant which can talk to you, guide you and automate things for you. So, doctors now no longer need to manually type the prescriptions. Just say it.

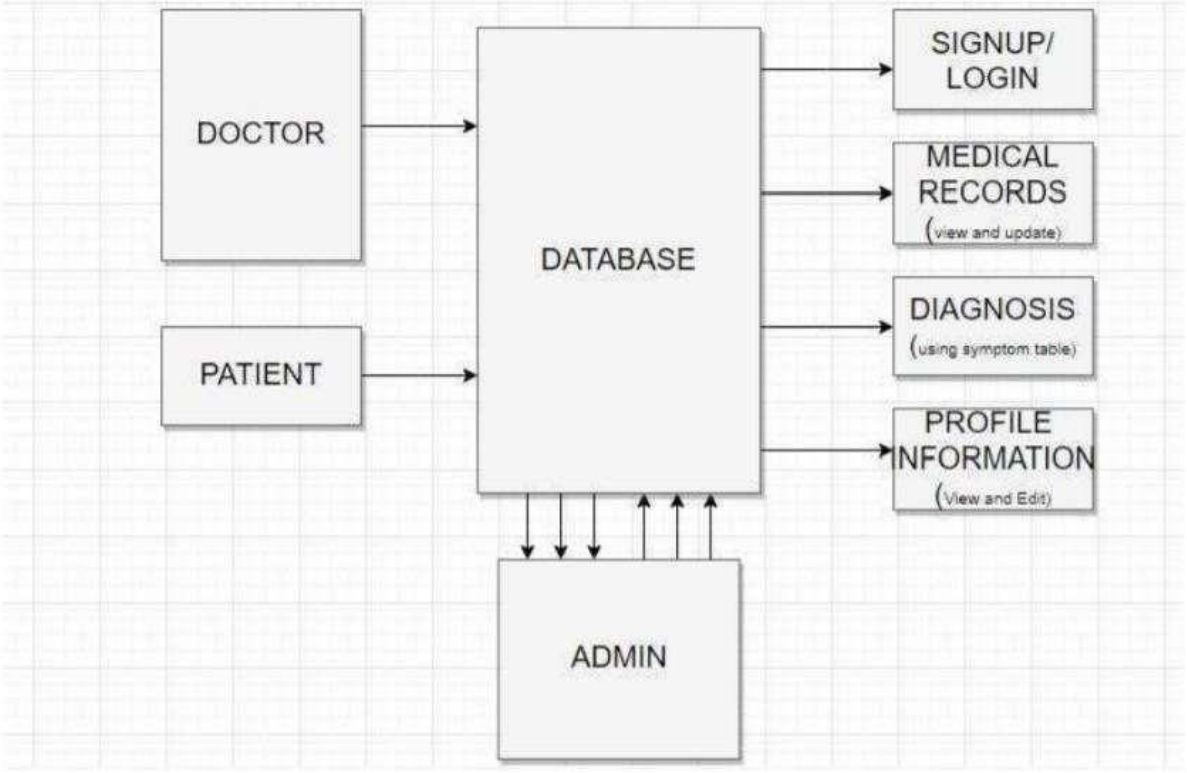



*Figure 1. Proposed Model*

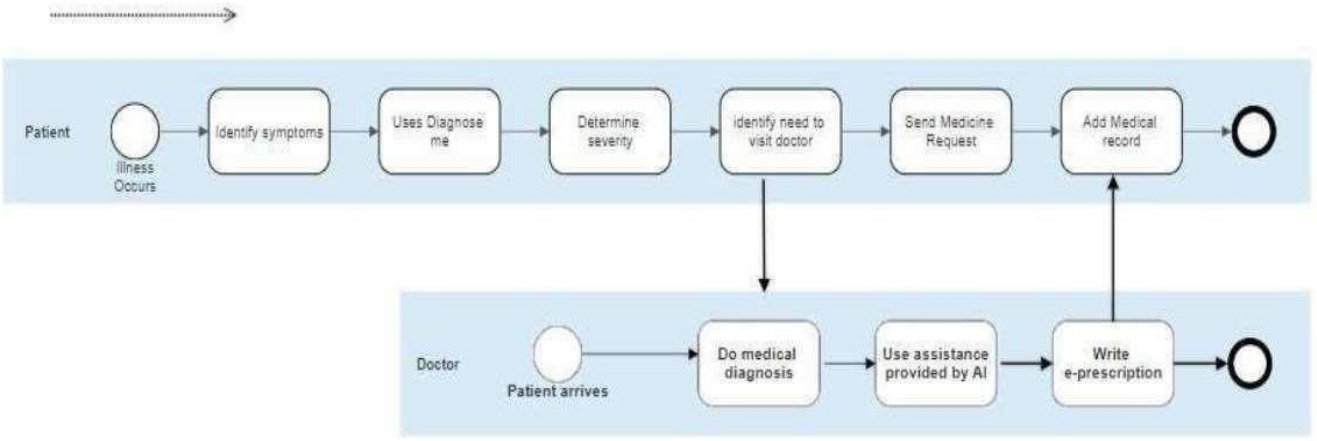



*Figure 2. State transition diagram for doctors and patients*

In this application, we have developed a platform to assist physicians with their diagnostic and patient-centered diagnostic tasks. We bring automation and digitalization to medical care with concepts such as Learning Machines, DBMS, AI and NLP.

This application is only intended to assist physicians and patients with the aim of providing medical care in an interactive, life-saving and cost-effective manner. This application is a small step towards achieving a larger target digitalizing completely our medical care.

## Literature Review

Collective prediction of diseases associated miRNAs based on transduction learning is being carried in [1]. This paper uses dataset HMDD v2.0: It is a database for experimentally supported microRNA of humans and disease associations. It mainly uses transduction leaning and performance measures AUC, precision, recall and F1-score. [2] Combined Supervised and Unsupervised Learning for improved miRNA target prediction. They use datasets from TCGA database. Combination of supervised and unsupervised approaches and is evaluated using conservation score, alignment score and energy. Major limitation in this proposed method is that the number of binding and network features were limited. In [3], relational regularized discriminative sparse learning for alzheimer's diseases diagnosis is used on ADNI database. Discriminative learning technique are used here and the classifications of AD versus NC, MCI versus NC, and pMCI versus sMCI is used for validation. In [4] Liver disease prediction using SVM and naïve bayes algorithms is being carried out using UCI repository dataset. Accuracy measures like accuracy, precision, TP rate, F measure and execution time were the measures used for evaluation but more detailed

prediction techniques required. Further extension of this study is highly desirable to direct the investigations towards real-world datasets instead of just theoretical approaches and simulations.

[5] is a performance analysis of all the available classification algorithms on early detection of Liver diseases. ILPD dataset from UCI repository is used and C5.0 algorithm and CHAID algorithm used for learning. Specificity, sensitivity, precision, fpr, FNR are the measures of accuracy. More detailed prediction required. Future work will be about a deep neural network and the standard backpropagation algorithm with weight decay (l2- 9 regularization) or sparsity (l1-regularization). Furthermore, weighted -Naïve Bayesian (WNB) algorithm will be applied with neural network and genetic algorithms. In [6] they are comparing performance of data mining algorithms in prediction of heart diseases. C5.0, KNN, SVM, neural network are used for learning on dataset UCI archive, built for machine learning. Specificity, Sensitivity, precision are measures of accuracy. Limitation is that more detailed prediction required. In [7] diseases are being predicted by machine learning over huge datasets from healthcare communities on hospital dataset. Naïve bayes, KNN, Decision trees are learning techniques used. Accuracy, precision, F measure, TPR and FPR are the performance measures used. But in this too more detailed prediction is required.

In [8] Survey of Machine Learning Algorithms for Disease Diagnostic is being carried out. UCI dataset used. BayesNet, SVM, decision tree are learning methods used with accuracy as performance measure. In [9], Improved Heart Disease Prediction Using Deep Neural Network. Cleveland dataset, UCI repository used and LSTM and Gated Neural Units used for learning. It has improved accuracy used with very feasible algorithm. In future, it can be extended to other applications of deep learning. [10] focuses on refining the accuracy for prediction of heart disease risk based on ensemble classification techniques. Cleveland dataset from UCI repository used. Bayes Net, Naïve Bayes, Random forest, PART uses for performance measures. Improving accuracy by implementing various algorithm should have been major focus. In [11], Comparing different supervised machine learning algorithms for disease prediction. Dataset from UCI repository used for logistic regression, SVM, Decision Tree, Random Forest, Naïve Bayes, KNN algorithms with accuracy used for performance measure. Implementing multiple algorithms together is a problem. In [12], Diabetes Disease prediction using Neural Network is being done on Cleveland dataset from UCI repository. ReLu activations function plays major role but more detailed prediction required

## PROPOSED METHODOLOGY

We are intending to actualize Random Forest algorithm to get the best and most effective outcomes. This calculation isn't commonly utilized in medicinal services based as per our exploration we can utilize it to achieve the best results.

In addition, existing frameworks tend to exclude clinical records and clinical history stockpiling together yet we are proposing a typical stage where we can club these things together to get the most efficient results as the clinical records and history can be additionally examined and later on. We can even implement Machine learning calculations on this information likewise which will make our outcomes significantly progressively precise and we can consider transforming our concentration into nonexclusive medications. Generally, we are proposing a complete clinical assistant.

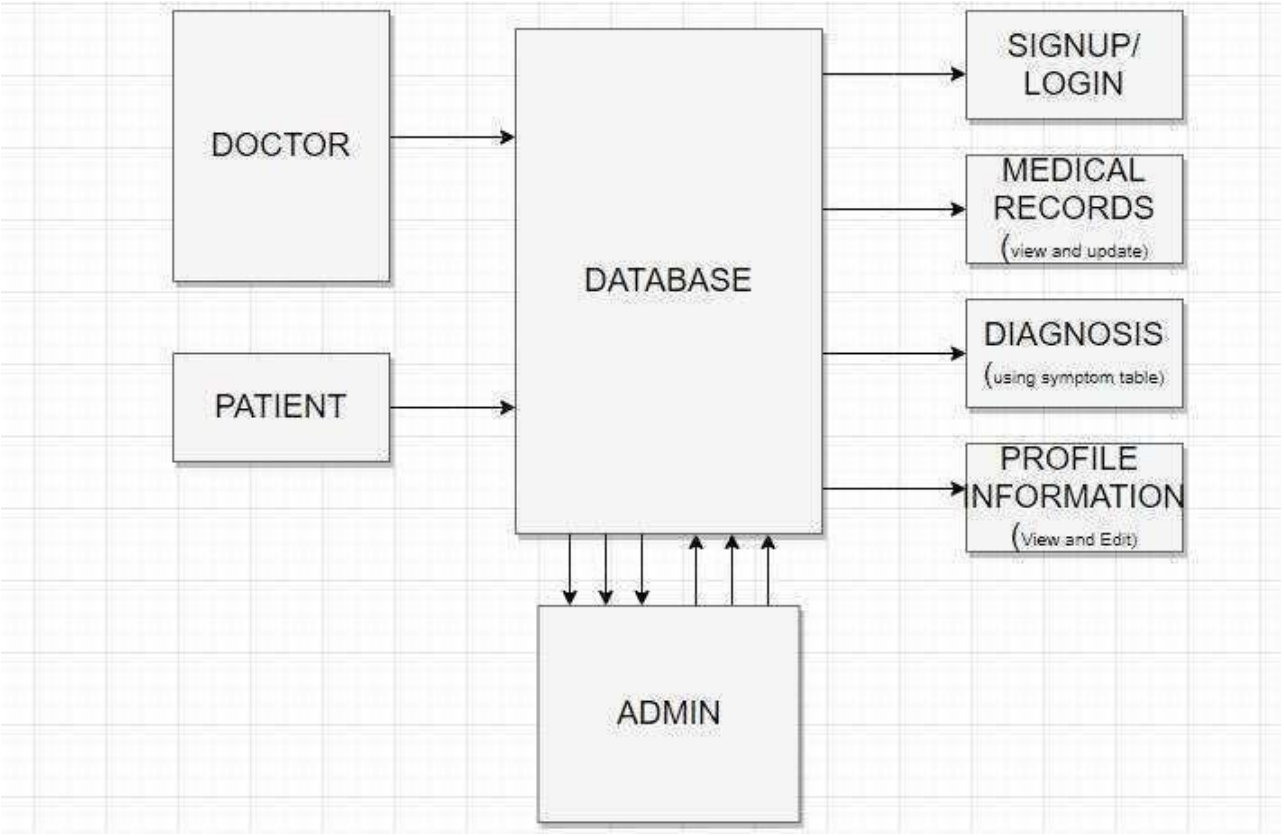



*Figure 3. System architecture*

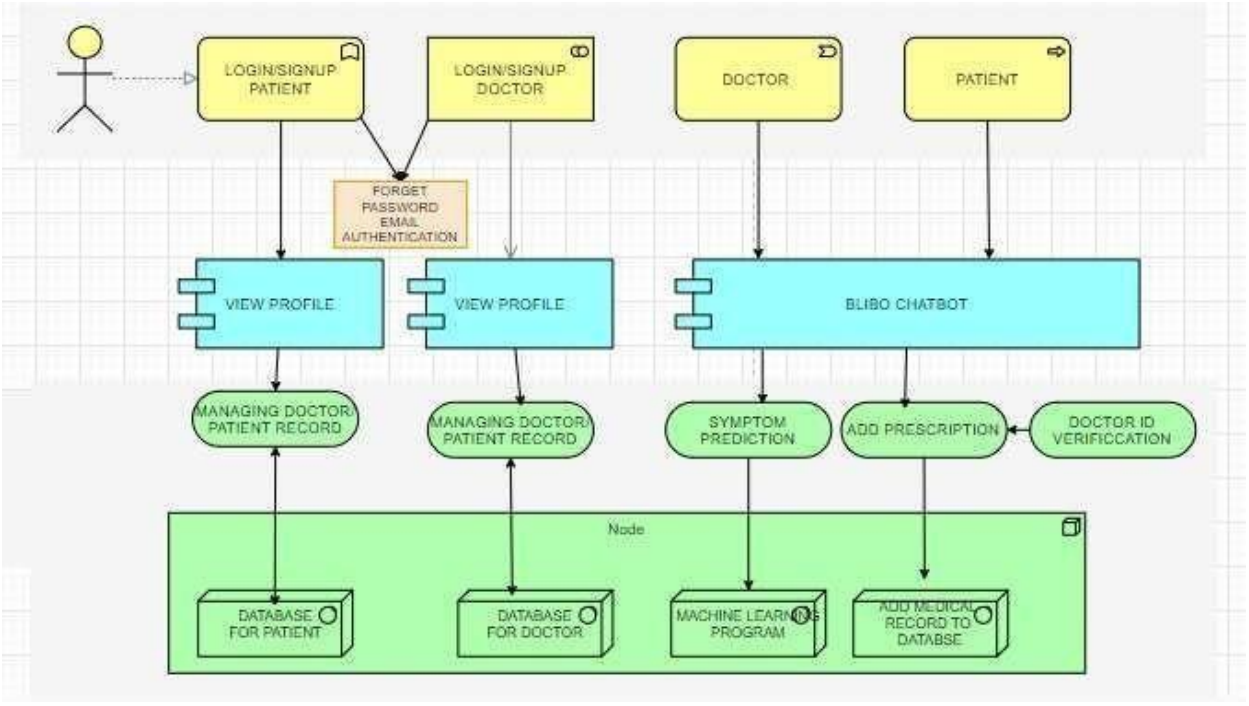



*Figure 4. System framework and design*

## IMPLEMENTATION

We oversee two databases for doctors and patients as they have various fields that give verification. Login/Signup module and Edit Profile Module depend on Set/Update and Creation of new records in the database that oversees and manages records. The symptom prediction module requires a machine learning code that contains all the potential side effects identified with basic issues that are often misdiagnosed in self-diagnosis. We additionally anticipate the likelihood of accurate prediction based on the number of manifestations coordinated and the number of illnesses with similar symptoms. This entire application can be run via our interactive Chatbot “BLIBO” which interacts with patient and doctor to help write prescription, check symptoms and edit profile hands free.

We have implemented this system using Flask back-end framework. We used flask as it uses python as programming language. It is a micro framework which allowed our system to be modular in design. Modularity ensures better and more manageable workflow and maintenance.

This system expects two kinds of users. A user can either be a patient or a doctor. So, we constructed a database system using ORM name Flask SQLAlchemy. This ORM is used as for create database which is SQL lite for now. We have two tables for patient and doctor which stores all the information regarding them. Another table named records is being used to store records for patients.

Account for user is created in Sign-up page. Here details like email and password are taken from the user. Password is hashed and stored in database to ensure it is stored safely.

Authentication system is managed by a Flask package called Flask-Login. User has to give email password to verify his identity. The password is converted in hash and matched with the hashed password stored in database. Selenium is used for automations.

Random Forest Classifier is used for disease prediction. The Random Forest follows certain tree planting rules, tree combinations, self-examination and post-processing, is extremely robust and is considered to be highly stable in the presence of vendors and in higher-sized facilities than other machine learning algorithms. The most important variable determination is the excluded selection made by RF through the informal funding process and tested by the Gini cut index. The Gini Index is a measure of the ability to predict variables in restorations or subdivisions, based on the principle of noise reduction; it is non-parametric and therefore doesn't depend on data which belongs to a particular type of distribution. The Gini index for a particular node is calculated as follows:

$$Gini(n)=1-\sum 2j=1(pj)2Gini(n)=1-\sum j=12(pj)2$$

where, $p_j$ is relative frequency of class $j$ in the node $n$.

For splitting a node in the best possible way, the development of the Gini index

should be enhanced. Next, a lower Gini value (e.g., a significant decrease in Gini) means that a particular predictive factor plays a major role in classifying data into two categories. Therefore, the Gini index was used to measure the significance of the segregation problem factors.

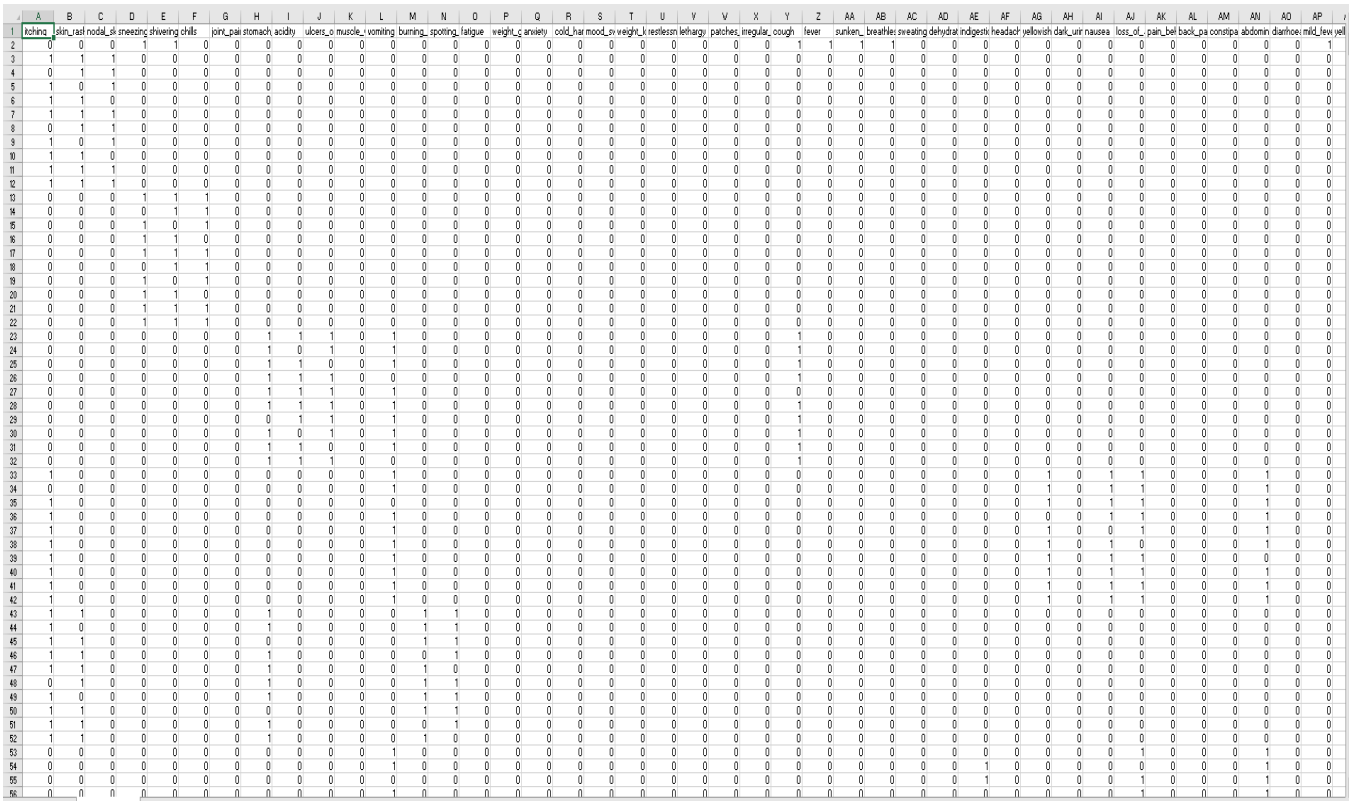
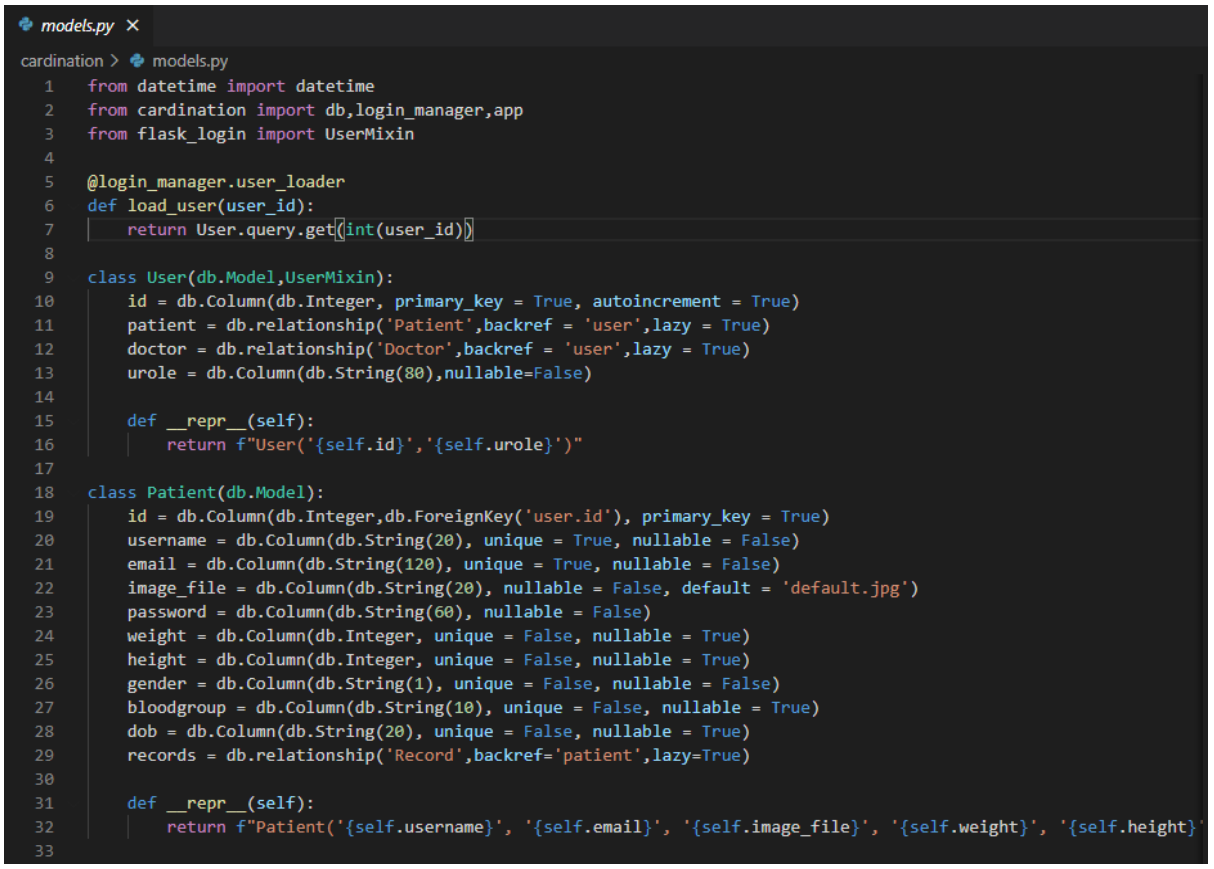


*Figure 5. Dataset screenshot*

We have analyzed over 4900 entries, 4922 to be precise and trained our machine learning model with such a huge dataset in order for it to provide us with the most accurate predictions. Random forest is an ensemble classifier, which constructs a group of independent and non-identical decision trees based on the idea of randomization. Each decision tree uses a random vector as a parameter, randomly selects the feature of samples, and randomly selects the subset of the sample data set as the training set.

# Code Screenshots and Information

**Models.py:** This file contains the structure of every table of our database. Our database consists of 4 tables. User, Patient, Doctor and Record table. User table is used for logins, Patient and Doctor tables stores the information of patient and doctor respectively. Record table stores the record of every user. Flask SQLAlchemy is used as the database ORM.



```python
class Doctor(db.Model):
    id = db.Column(db.Integer, db.ForeignKey('user.id'), primary_key = True)
    username = db.Column(db.String(20), unique = True, nullable = False)
    email = db.Column(db.String(120), unique = True, nullable = False)
    image_file = db.Column(db.String(20), nullable = False, default = 'default.jpg')
    password = db.Column(db.String(60), nullable = False)
    specialization= db.Column(db.String(60), nullable = False)
    degree= db.Column(db.String(60), nullable = False)
    gender= db.Column(db.String(1), nullable = True)
    phone_number= db.Column(db.String(60), nullable = False)
    address= db.Column(db.String(120), nullable = True)
    records = db.relationship('Record',backref='doctor',lazy=True)

    def __repr__(self):
        return f"Doctor('{self.username}', '{self.email}', '{self.image_file}', '{self.specialization}', '{self.c

class Record(db.Model):
    id = db.Column(db.Integer,primary_key=True,autoincrement=True)
    date = db.Column(db.String(30),nullable=False,default=datetime.utcnow)
    symptoms = db.Column(db.String(20),nullable=True)
    disease = db.Column(db.String(20), nullable=False)
    prescription = db.Column(db.String(20),nullable=False)
    additional_info = db.Column(db.Text)
    patient_id = db.Column(db.String(20),db.ForeignKey('patient.id'),nullable=False)
    doctor_id = db.Column(db.String(20),db.ForeignKey('doctor.id'),nullable=False)

    def __repr__(self):
        return f"Record('{self.id}', '{self.date}', '{self.symptoms}','{self.disease}','{self.prescription}','{se
```

**Voice assistance:** Voice assistant is made in python using machine learning. Voice assistant has automation capabilities used for filling up forms and ordering medicines from third party applications like netmeds. Automation is done using selenium web driver. Random forest classifier is used for disease predictions. Text to speech is done using JavaScript libraries.

**Utils.py- 172 Lines of code.**

```python
line_num = -1
flag = False
all_symptom = ['itching','skin_rash','nodal_skin_eruptions','sneezing','s
diagnosis = False
df = pd.read_csv('Training.csv')
df.head()
X = df.iloc[:, :-1]
y = df['prognosis']
rf_clf = RandomForestClassifier()
rf_clf.fit(X, y)
```

```
def predict(sym):
    global rf_clf,X,Y
    symptoms_dict = {}
    for index, symptom in enumerate(X):
        symptoms_dict[symptom] = index
    input_vector = np.zeros(len(symptoms_dict))
    for i in sym:
        input_vector[[symptoms_dict[i]]] = 1
    rf_clf.predict_proba([input_vector])
    pred = rf_clf.predict([input_vector])
    return str(pred[0])
```

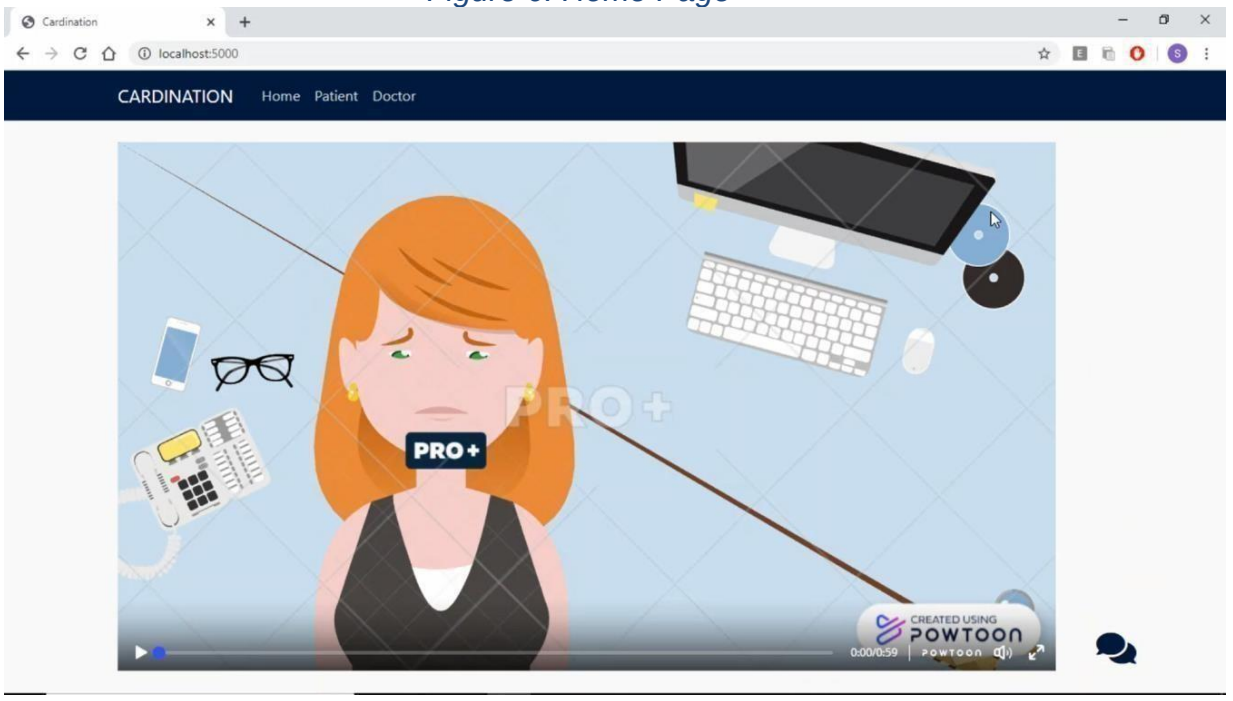
```
def searching(mes):
    global flag, fillform, all_symptom
    global line_num, counter, diagnosis
    h= open("sentences.txt","r")
    line = h.readlines()
    num_of_line = len(line)
    mes = mes.lower()
    if mes.find("purchase")!=-1:
        med = mes.split("purchase")[1].strip()
        browser = webdriver.Chrome("C:\\Users\\sarth\\Desktop\\chromedriver.exe")
        browser.maximize_window()
        browser.get("https://www.netmeds.com/")
        search_bar = browser.find_element_by_xpath("//*[@id='search']")
        search_bar.send_keys(med)
        search = browser.find_element_by_xpath("//*[@id='search_form']/button")
        search.click()
        browser.implicitly_wait(30)
        return "Processing order request from netmeds"

    if mes == "run diagnosis":
        diagnosis = True
        return "Give comma separated symptoms which you are facing."
    if diagnosis==True:
        symptoms = mes.split(",")
        simp = []
        for i in range(len(symptoms)):
            symptoms[i] = (symptoms[i].strip()).replace(" ", "_")
            if symptoms[i] in all_symptom:
                simp.append(symptoms[i])
        result = predict(simp)
```

# Implementation

**Home page:** First Landing page of our website. Contains an explanatory video which presents the functions presented by our software.

*Figure 6. Home Page*




## Patient Module:

One of the two modules of our software made for providing easy user interaction to the patient. Contains all the patient specific features and a voice assistant to help them out at any point in time.

1. **Sign Up portal:** Patient enters all the basic medical information and a profile is created for every patient. Medical information contains DOB, Blood group, Height, Weight, Age etc. Email and password have to be set by the patient. Email can be used for password recovery in the future too.

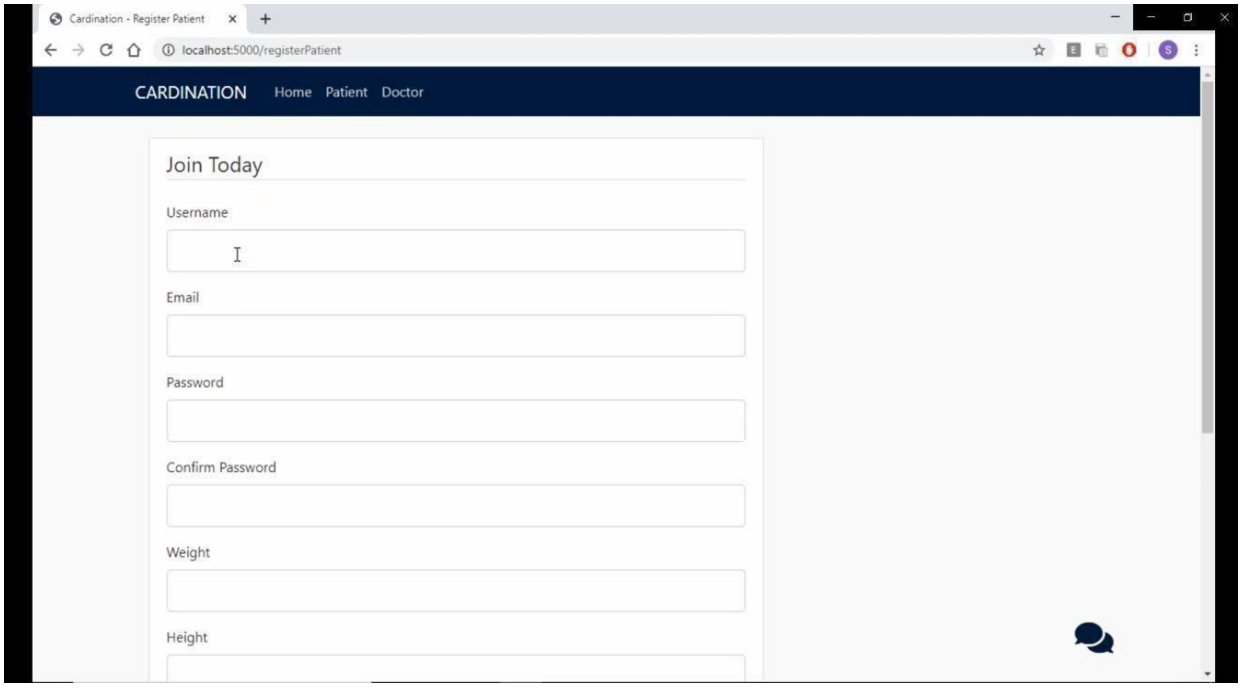



*Figure 7. Sign-up page*

1. **Login Portal:** Email and password are authenticated and if correct the patient is authorized to view the content presented.

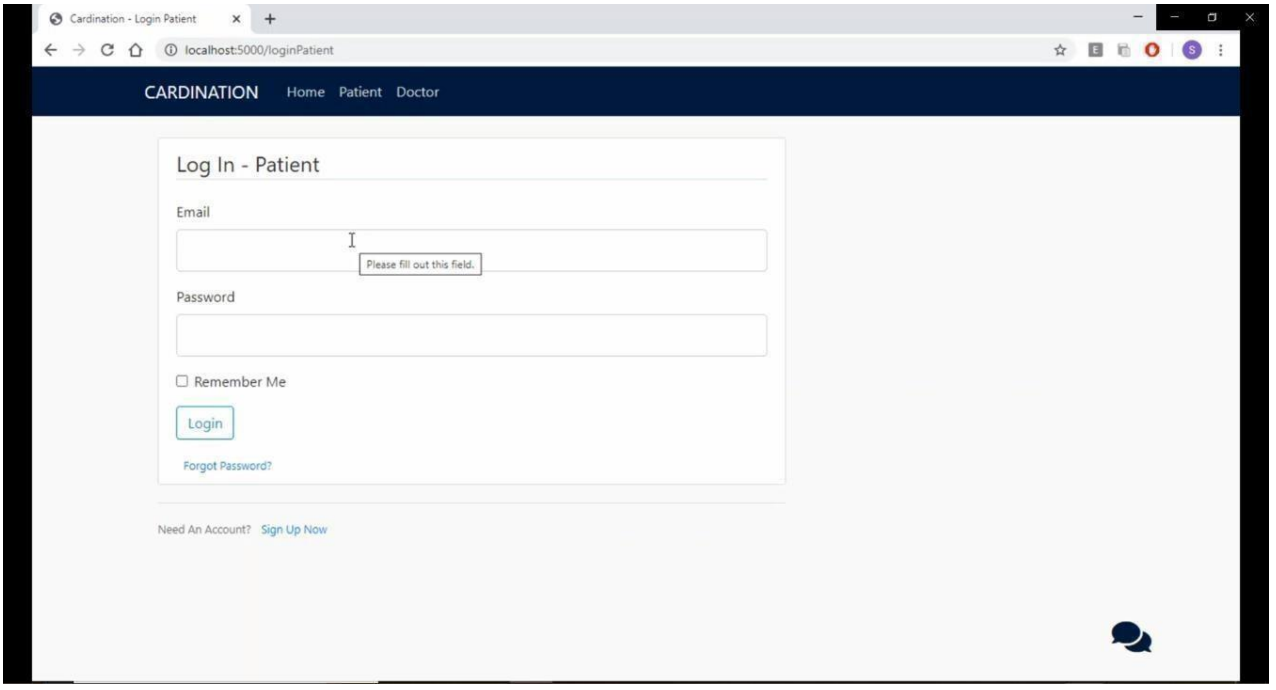



*Figure 8. Login Page*

2. **Patient records:** Medical records of patients stored in order of date. Doctor, disease medication and additional information can be viewed on this page. Information can be sorted and searched.

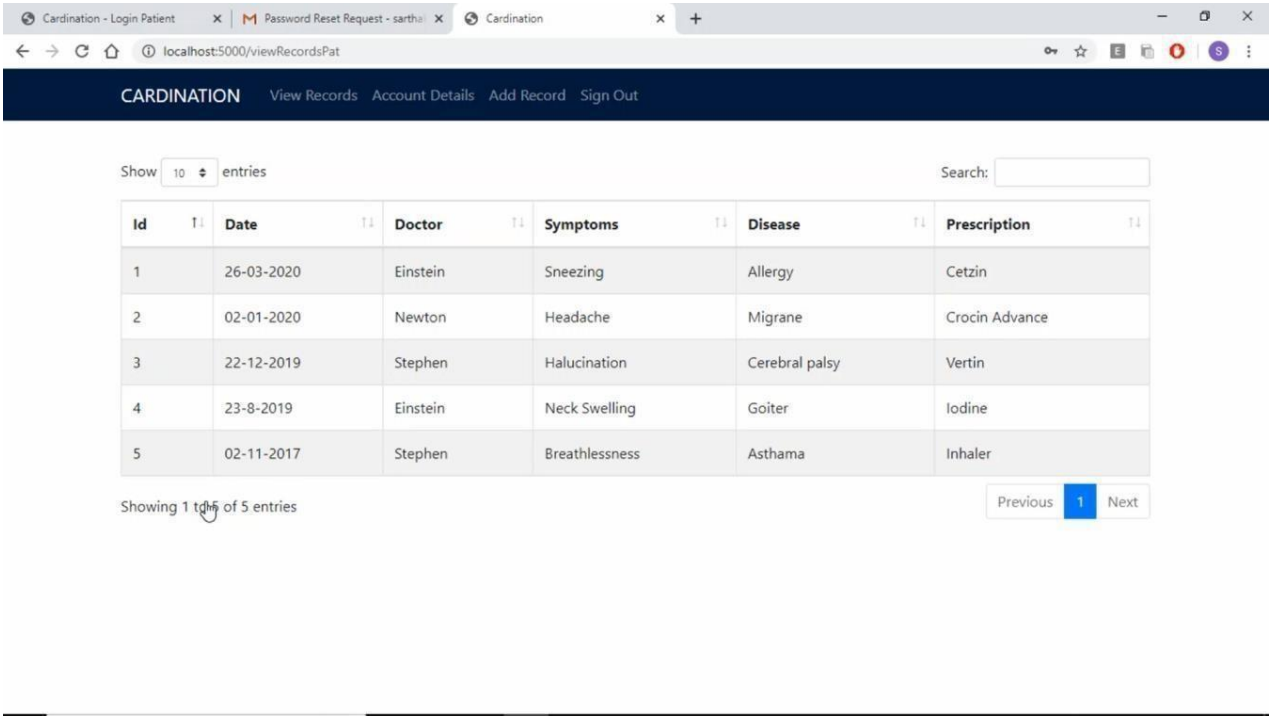



*Figure 9. Patient Records*

3. **Patient Profile:** Patient can view and edit his or her profile details set up a profile photo for his account.

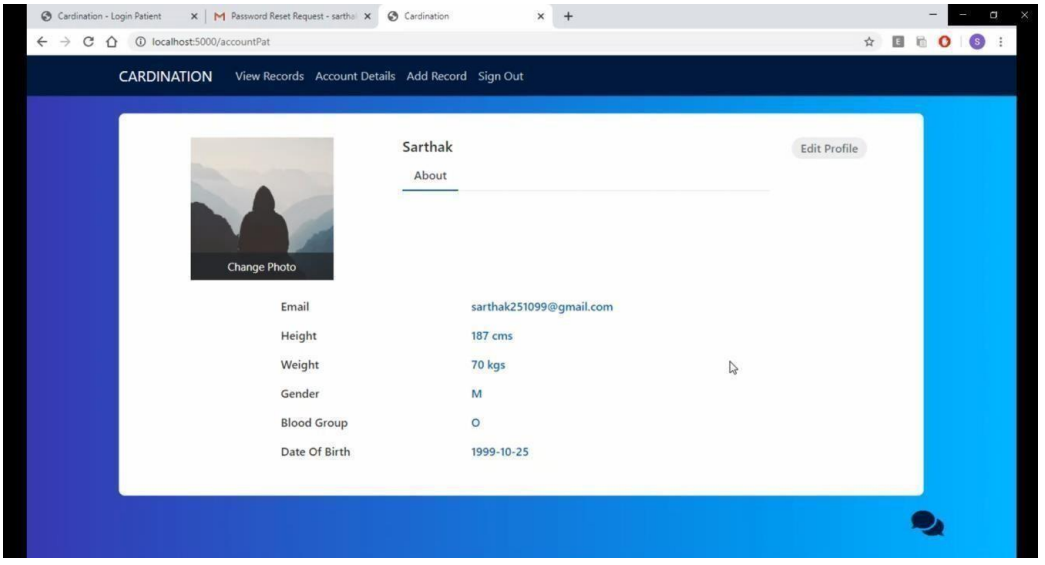



*Figure 10 Patient Profile*

## Doctor Module:

Second module of our software made for doctors. Doctors has some doctor specific features in their view and a disease prediction software to help them out in the process of disease prediction.

1. **Login Portal:** Authenticate and Authorize the doctor by taking in email and password. Also provides a forgot password link for password recovery through registered mail provided by the doctor.

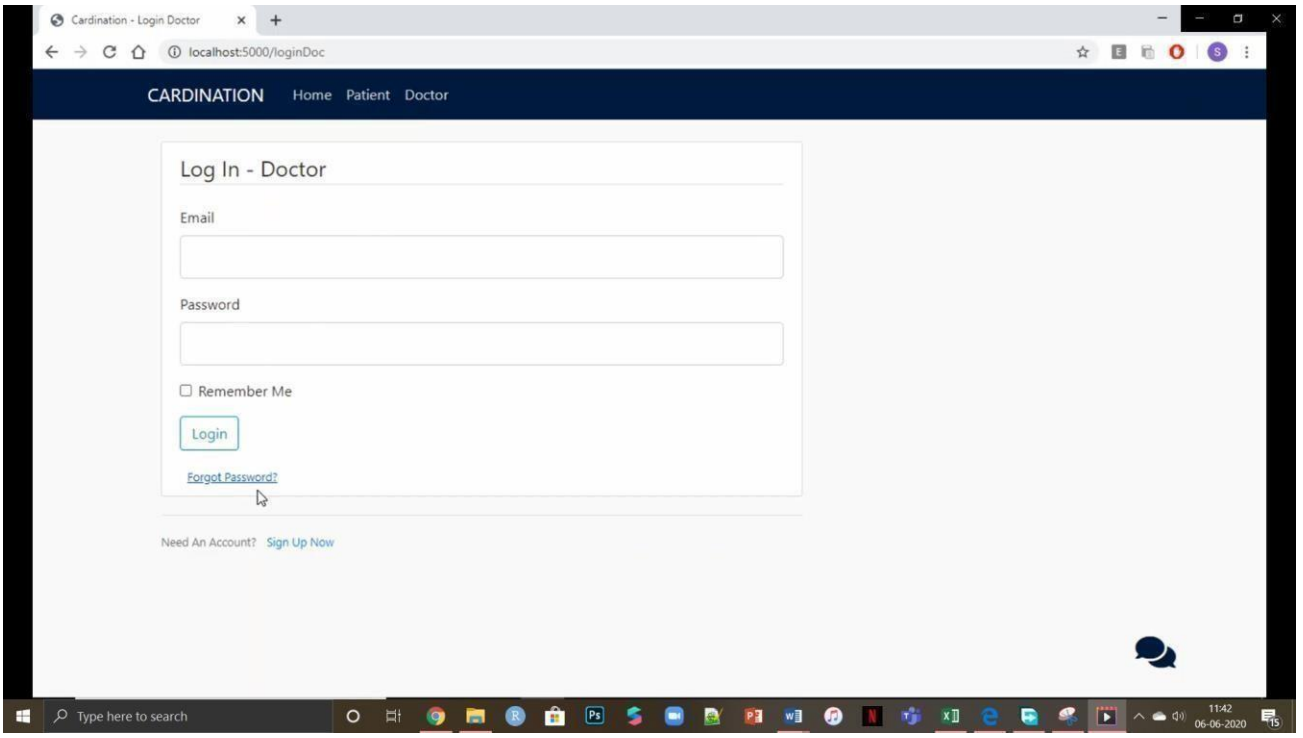



*Figure 11. Doctor Login*

2. **Voice Assistant:** The key feature of our software provided to both patient and doctor. This provide a hands-free assistance to the users to help them in any point of time. It's difficult to understand a new system for novice users. This voice assistance can help them by automating almost any task for them. Just ask it to do something for you and it will do it for you. It has added functionality for helping in disease diagnosis using machine learning prediction model. It can fill up forms, add and delete records, order medicines or simply chat with you.

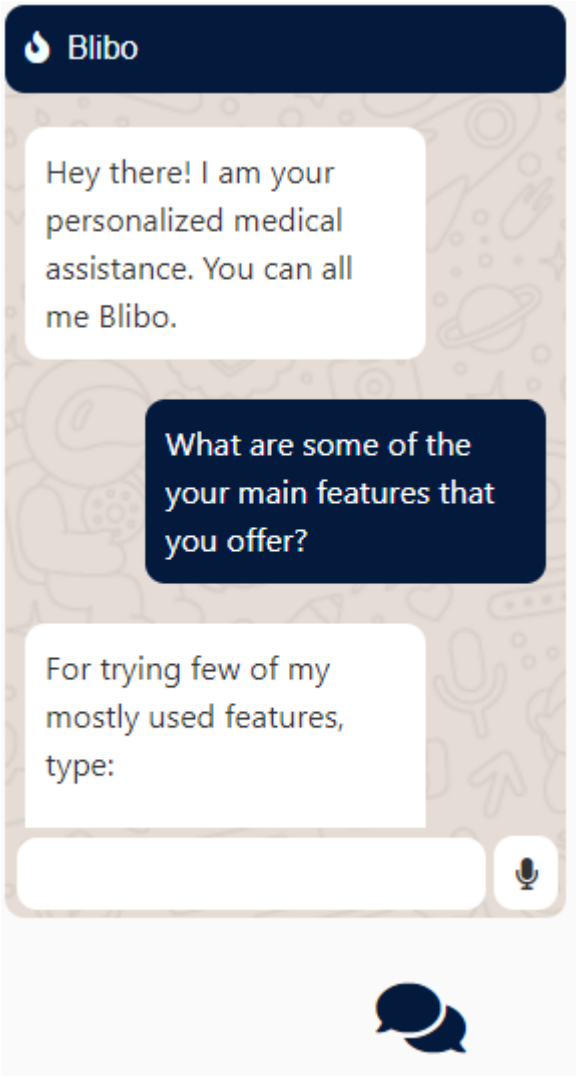



*Figure 12. Chatbot screenshot*

# Testing and Validation:

## TU0-Test case 0:

**Title:** Login Page

**Description:** A registered user should be able to successfully login to system.

**Precondition:** The user must already be registered with an email address and password.

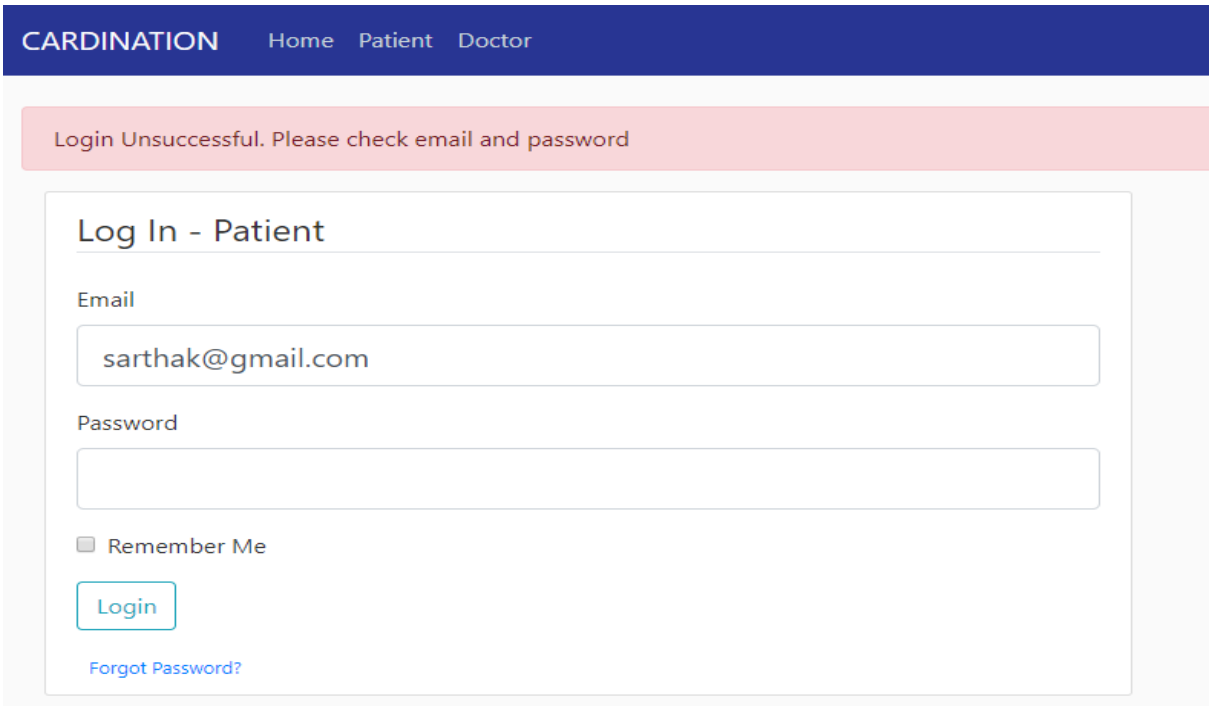
**Assumption:** A supported browser must be used.

**Test Steps:**

- In the email field enter the user's email address.
- Enter the password of user.
- Click on 'Sign In'.

| Test case ID | Test Data | Expected Result | Actual Result | Pass/Fail |
|---|---|---|---|---|
| TU01 | Email: '*sarthak251099@gmail.com*' Password: '*Password*' | User must be logged in | User logged in successfully | Pass |
| TU02 | Email: 'sarthak@gmail.com' Password: 'sarthak' | User must not be logged in. | User denied access. | Pass |

**Invalid Login:**



## TU1-Test case 1:

**Title:** Sign up Page

**Description:** An unregistered user should be able to create his account by giving the necessary details to be able to login later.

**Precondition:** None.

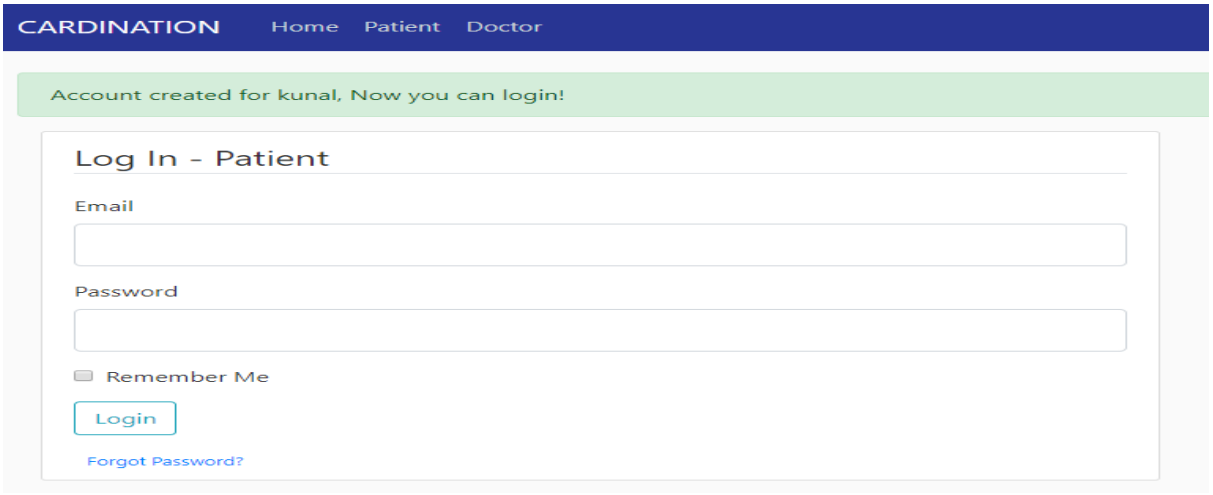
**Assumption:** A supported browser must be used.

**Test Steps:**

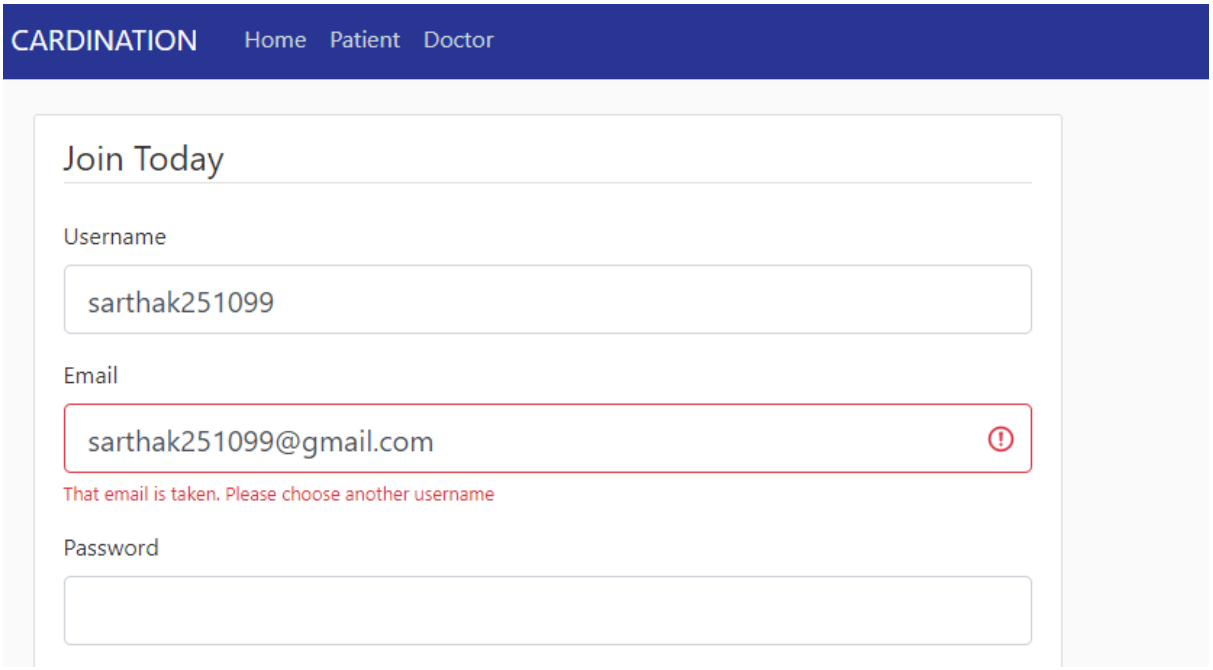
- In the email field enter the user's email address.
- Enter the password of user.
- Confirm the password for the user.
- Enter all other details.
- Click on 'Sign Up'.

| Test case ID | Test Data | Expected Result | Actual Result | Pass/Fail |
|---|---|---|---|---|
| TU11 | Email: '*dhruv@gmail.com*' Password: '*pass*' | Account must be generated | Account generated for the user. | Pass |
| TU12 | Email: '*sarthak251099@gmail.com*' Password: '*sarthak*' | Should raise error saying account already exists. | Error raised 'account already exists, | Pass |
| TU13 | Email: '*sarthak251099*' Password: '*surana*' | Incorrect email address error displayed. | Incorrect email error displayed. | Pass |

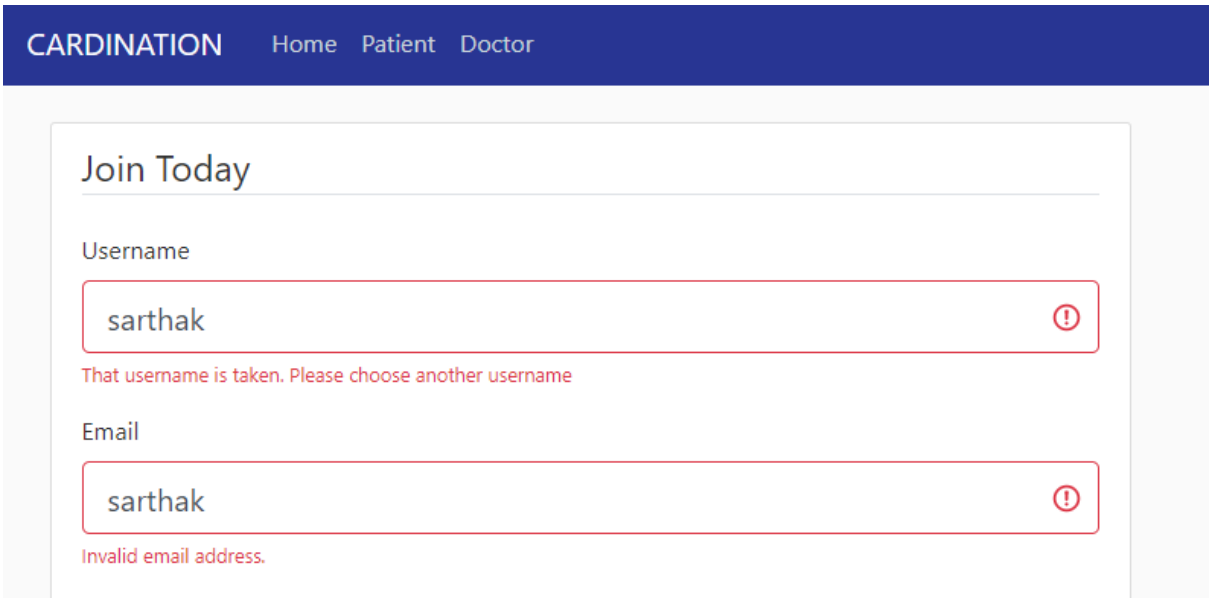
**TU11 Screenshot:** *'Account created successfully'*



**TU12 Screenshot:** *'email already exists'*




**TU12 Screenshot:** *'Invalid email address'*




## TU2-Test case 2:

**Title:** View medical record page.
**Description:** A logged in user should see his medical history and should be able to search a particular medical history and sort history on any criteria.

**Precondition:** User must be logged in.

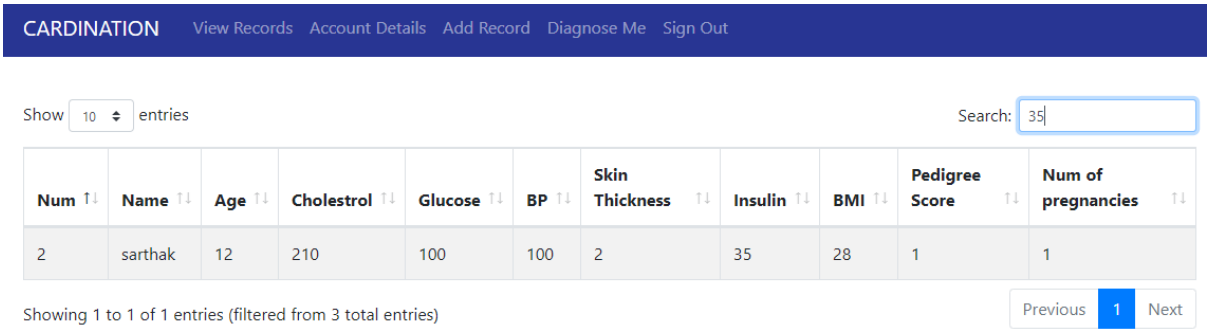
**Assumption:** A supported browser must be used.

**Test Steps:**

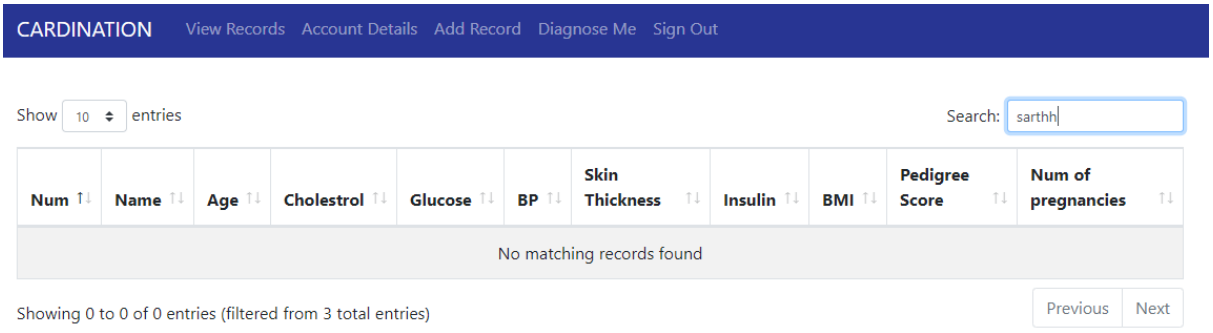
- Type any medical history related item in search bar.
- Press enter to search for that item.

| Test case ID | Test Data | Expected Result | Actual Result | Pass/Fail |
|---|---|---|---|---|
| TU21 | Search: '35' | Entry with 35 as value should be displayed. | Entry with 35 as value displayed | Pass |
| TU22 | Search: 'sarthh' | Should not show any entry as none of them has 'sarthh' | No entry shown | Pass |

**TU21 Screenshot:** *Entry with value 35 shown.*




**TU21 Screenshot:** *No entry displayed.*



## TU3-Test case 3:

**Title:** Adding medical record.

**Description:** A logged in user should be able to add his medical record.

**Precondition:** User must be logged in.

**Assumption:** A supported browser must be used.

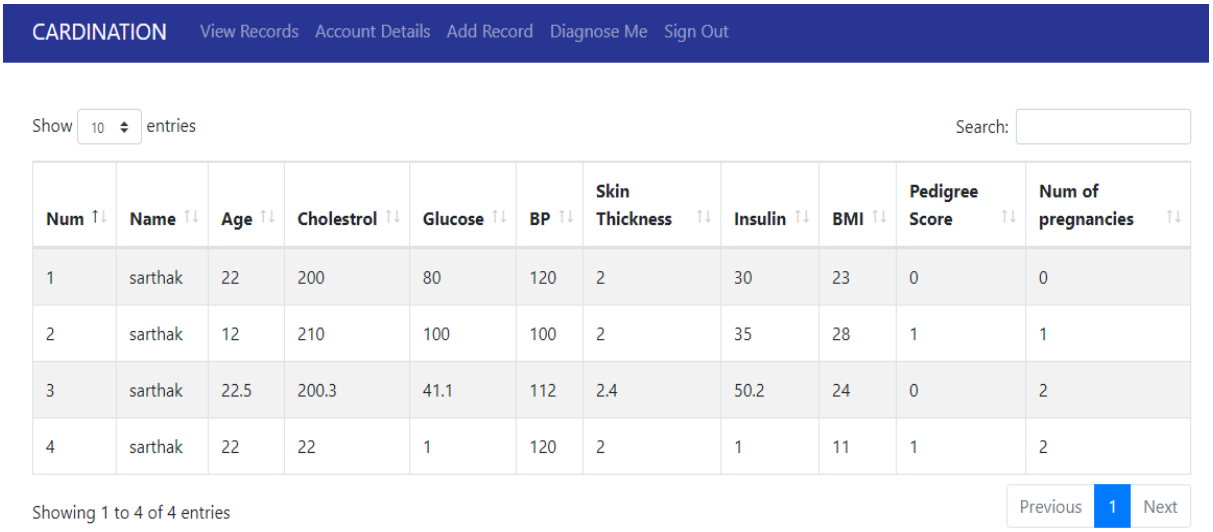
**Test Steps:**

- Write down in all the fields.
- Click on 'Add Record' button at the bottom.

| Test case ID | Test Data | Expected Result | Actual Result | Pass/Fail |
|---|---|---|---|---|
| TU31 | Enter all the details with its specified data type. | New record should be added and displayed. | New record successfully added and displayed. | Pass |
| TU32 | Entering invalid datatype in one field. | Raise error | Error raised | Pass |
| TU33 | Leaving a field empty and pressing add record. | Empty field error should be displayed. | Error displayed on the page. | Fail |

**TU31 Screenshot:** *All valid entries.*

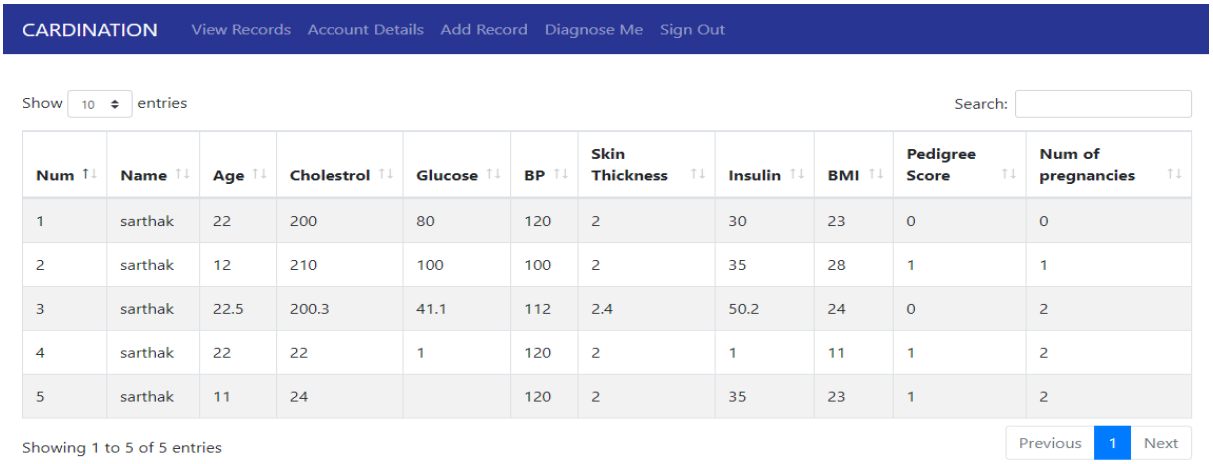

Record added successfully.




**TU32 Screenshot:** *One or more invalid entries.*


Error raised.


**TU33 Screenshot:** *Leaving one or more required fields empty.*




Entry made without showing error.

## TU4-Test case 4:

**Title:** Diagnose Me

**Description:** A user should be able to diagnose his illness using the diagnose me functionality provided.

**Precondition:** User must be logged in.

**Assumption:** A supported browser must be used.

**Test Steps:**

- Say diagnose me.
- Say all the symptoms being faced.
- Say done after giving the symptoms.

| Test case ID | Test Data | Expected Result | Actual Result | Pass/Fail |
|---|---|---|---|---|
| TU41 | Diagnose me:<br>*'Fever'*<br>*'Cough'*<br>*'Sneezing'* | Illness should be predicted and shown. 'You might have Bronchial Asthma' | 'You might have Bronchial Asthma' displayed correctly. | Pass |
| TU42 | Entering invalid symptoms (symptoms which are not existing in the dataset).<br>*'High fever'*<br>*'Watery eyes'* | Symptoms not recognised. Enter valid symptoms | No error shown on the frontend. No illness predicted. Fatal error in the code. | Fail |

# Demo Video Link

https://drive.google.com/file/d/1U_oK_PUsYShTRzTZTZFJwe0dXXj71lL4/view?usp=sharing

# POSTER

## Doctor's Assistant System

SARTHAK SURANA, DHRUV MITTAL | ARUN KUMAR G | SCOPE

### Introduction

Self-diagnosis is highly susceptible to error and is often extremely dangerous if inappropriate decisions are made on the basis of a misdiagnosis. After observing these misdiagnosis activities and wrong medications by practitioners, we thought of building a web application for the welfare of our society.

This web app can help doctors and patients alike to store Electronic medical records, help in the process of disease diagnosis, fasten the process of prescription generation, aid in buying prescription medicines from 3rd party websites and much more.

### Motivation

A doctor writes a prescription for you, but you aren't able to understand it? Or you wake up in the middle of the night with pain near the chest, do you immediately take medicine for heart ? It might've just been a gastric pain? These problems are some common scenarios in medical field that we face in day to day life.

### SCOPE of the Project

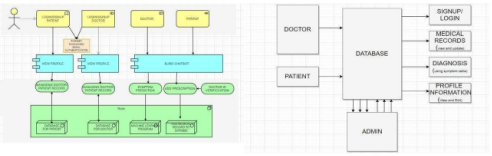
We will use the **incremental model** for software development as I have my projects requirements clear, like I have 4 modules to work on which are **The Login and signup module which also has the overall GUI for the software, Doctor's side(chatbot module to assist his tasks), the patient record module( which stores the past medical records) and the medicine prediction module to help oblivious patients.** Moreover, all these options are independent of each other and I can prioritize the requirements (increments) to be delivered orderly.

### Methodology

We are intending to actualize Random Forest algorithm to get the best and most effective outcomes. This calculation isn't commonly utilized in medicinal services based as per our exploration we can utilize it to achieve the best results.

In addition, existing frameworks tend to exclude clinical records and clinical history stockpiling together yet we are proposing a typical stage where we can club these things together to get the most efficient results as the clinical records and history can be additionally examined and later on. We can even implement Machine learning calculations on this information likewise which will make our outcomes significantly progressively precise and we can consider transforming our concentration into nonexclusive medications. Generally, we are proposing a complete clinical assistant.

We oversee two databases for doctors and patients as they have various fields that give verification. Login/Signup module and Edit Profile Module depend on Set/Update and Creation of new records in the database that oversees and manages records. The symptom prediction module requires a machine learning code that contains all the potential side effects identified with basic issues that are often misdiagnosed in self-diagnosis. We additionally anticipate the likelihood of accurate prediction based on the number of manifestations coordinated and the number of illnesses with similar symptoms.




This entire application can be run via our interactive Chatbot "BLIBO" which interacts with patient and doctor to help write prescription, check symptoms and edit profile hands free.

We have implemented this system using Flask back-end framework. We used flask as it uses python as programming language. It is a micro framework which allowed our system to be modular in design.

This system expects two kinds of users. A user can either be a patient or a doctor. So, we constructed a database system using ORM name Flask SQLAlchemy. This ORM is used as for create database which is SQL lite for now. We have two tables for patient and doctor which stores all the information regarding them. Another table named records is being used to store records for patients.

Authentication system is managed by a Flask package called Flask-Login. User has to give email password to verify his identity. The password is converted in hash and matched with the hashed password stored in database.

### Results

While working on this app, our main motive was to help the doctors as well as patient with the medical assistance. We wanted to help the patients with their medical record management so that they can store their previous medical records and prescriptions safely and never lose them in future, which in turn can be used by the doctors to refer and provide the best suitable prescription to the patient.

So were able to produce a web app which could manage their data efficiently and storing the login details in hash which made the system more secure.

To make the addition of record much easier, we introduced a Voice Assistant named "BLIBO" which will help you to fill up the forms via voice making all process hassle free. Also the voice assistant will help you purchase the medicines via redirecting you to the medicine E-commerce web page via automations.

For those doctors, who are not 100% sure of predicted diagnosis through the symptoms and need assistance for predicting disease, we have a solution for that too. Our voice assistant is capable to predict disease using multiple Machine Learning models and algorithms, taking the symptoms faced by the patients as the input, and chose the best model which had the maximum accuracy for each and every test cases.

Since each machine learning model has some pros and cones, and they work well with certain type of test cases, we were able to land with an outcome which would work with almost all types of test cases in form of symptoms in overall.

Hence, the doctors will be able to diagnose the patient with 100% accuracy and provide the best suitable advice and prescription based on the previous medical history and records if the patient already suffered through a similar disease.

### Conclusion

This application is exclusively made to assist doctors and patient with intention of giving medical care in an interactive, life saving and resource saving manner. This application is a minor step in achieving a bigger target of completely digitalizing our medical care. A lot more work can be done to further improve our initiative in this regard such as video conferencing between patient and doctor, so that our doctors are one click away from us. We can improve our security substantially in terms of password protection and medication misuse. The world is digitalizing everything at a rapid pace, and the medical care sector, which has time and time proven to be the backbone of our lives, should not fall behind in this regard.

### References


- Ahmed K, Jesmin T, Rahman MZ (2013). Early prevention and detection of skin cancer risk using data mining.
- Int J Comput App, 62, 1-6. Amarathunga AA, Ellawala EP, Abeysekara GN, Amalraj CR (2015). Expert system for diagnosis of skin diseases.
- Int J Sci Technol Res, 4, 174-8. Bakpo FS, Kabari LG (2011). Diagnosing skin diseases using an Artificial Neural Network, in Artificial Neural Networks-Methodological Advances and Biomedical Applications; In Tech Publisher, pp 253- 70.
- Bojarczuka CC, Lopesb HS, Freitasc AA (2001). Data mining with constrainedsyntax genetic programming: applications in medical data set. Algorithms, 6, 7-0.
- Chang CL, Chen CH (2009). Applying decision tree and neural network to increase quality of dermatological diagnosis. Exp Sys with App, 36, 4035 – 41.
- Chaurasia V, Pal S, Tiwari BB (2018). Chronic Kidney disease: A predictive model using decision tree. Int JEng Res Technol, 11, 1781-94.
- Chaurasia V, Pal S, Tiwari BB (2018). Prediction of benign and malignant breast cancer using data mining techniques.

## Conclusion and future scope

In this application, we develop a platform to facilitate doctors with their jobs of identifying patient issues and patient to self-diagnose itself in case of emergency. We bring automation and digitalization to medical care through concepts like machine learning, DBMS, AI and NLP. This application is exclusively made to assist doctors and patient with intention of giving medical care in an interactive, life-saving and resource saving manner. This application is a minor step in achieving a bigger target of completely digitalizing our medical care.

A lot more work can be done to further improve our initiative in this regard such as video conferencing between patient and doctor, so that our doctors are one click away from us. We can improve our security substantially in terms of password protection and medication misuse. The world is digitalizing everything at a rapid pace, and the medical care sector, which has time and time proven to be the backbone of our lives, should not fall behind in this regard.

## REFERENCES


[1] Ahmed K, Jesmin T, Rahman MZ (2013). Early prevention and detection of skin cancer risk using data mining.

[2] Int J Comput App, 62, 1-6. Amarathunga AA, Ellawala EP, Abeysekara GN, Amalraj CR (2015). Expert system for diagnosis of skin diseases.

[3] Int J Sci Technol Res, 4, 174-8. Bakpo FS, Kabari LG (2011). Diagnosing skin diseases using an Artificial Neural Network, in Artificial Neural Networks - Methodological Advances and Biomedical Applications; In Tech Publisher, pp 253- 70.

[4] Bojarczuka CC, Lopesb HS, Freitasc AA (2001). Data mining with constrainedsyntax genetic programming: applications in medical data set. Algorithms, 6, 7-0.

[5] Chang CL, Chen CH (2009). Applying decision tree and neural network to increase quality of dermatological diagnosis. Exp Sys with App, 36, 4035 – 41.

[6] Chaurasia V, Pal S, Tiwari BB (2018). Chronic Kidney disease: A predictive model using decision tree. Int JEng Res Technol, 11, 1781-94.

[7] Chaurasia V, Pal S, Tiwari BB (2018). Prediction of benign and malignant breast cancer using data mining techniques.

[8] Fatima, Meherwar & Pasha, Maruf. (2017). Survey of Machine Learning Algorithms for Disease Diagnostic. Journal of Intelligent Learning Systems and Applications. 09. 1-16. 10.4236/jilsa.2017.91001.

[9] Improved Heart Disease Prediction Using Deep Neural Network. Mohd Ashraf, M. A. Rizvi and Himanshu Sharma Volume 8 No.2 April-June 2019 pp 49-54

[10] Divya Jain, Vijendra Singh, Feature selection and classification systems for chronic disease prediction: A review, Egyptian Informatics Journal, Volume 19, Issue 3, 2018, Pages 179-189, ISSN 1110-8665, https://doi.org/10.1016/j.eij.2018.03.002.

[11] Uddin, S., Khan, A., Hossain, M. et al. Comparing different supervised machine learning algorithms for disease prediction. BMC Med Inform Decis Mak 19, 281 (2019). https://doi.org/10.1186/s12911-019-1004-8

[12] A deep learning approach based on convolutional LSTM for detecting diabetes. Motiur Rahman, Dilshad Islam, Rokeya Jahan, Mukti Indrajit Saha. https://doi.org/10.1016/j.compbiolchem.2020.107329